Package ‘RRreg’

December 4, 2019

Type Package

Title Correlation and Regression Analyses for Randomized Response Data

Version 0.7.1

Date 2019-12-04

Author Daniel W. Heck [aut, cre] (<https://orcid.org/0000-0002-6302-9252>), Morten Moshagen [aut]

Maintainer Daniel W. Heck <dheck@uni-marburg.de>

Depends R (>= 3.5.0)

Imports parallel, doParallel, foreach, stats, grDevices, graphics, lme4

Suggests knitr, R.rsp


License GPL-3

Encoding UTF-8

LazyLoad yes


VignetteBuilder knitr, R.rsp
**Description**

Univariate and multivariate methods for randomized response (RR) survey designs (e.g., Warner, 1965). Univariate estimates of true proportions can be obtained using `RRuni`. RR variables can be used in multivariate analyses for correlations (`RRcor`), as dependent variable in a logistic regression (`RRlog`), or as predictors in a linear regression (`RRlin`). The function `RRgen` generates single RR data sets, whereas `RRsimu` generates and analyzes RR data repeatedly for simulation and bootstrap purposes. An overview of the available RR designs and examples can be found in the package vignette by `vignette('RRreg')`.

**Details**

- **Package**: RRreg
- **Type**: Package
- **Depends**: R (>= 3.0.0)
- **Imports**: parallel, doParallel, foreach, stats, grDevices, graphics, lme4
- **Suggests**: knitr
- **License**: GPL-2
getPW

URL: https://www.sowi.uni-mannheim.de/en/erdfelder/research/software/rrreg/

Citation

If you use RRreg in publications, please cite the package as follows:

Author(s)

Daniel W. Heck <dheck@uni-marburg.de> and Morten Moshagen <morten.moshagen@uni-ulm.de>

References


See Also

Useful links:
- https://www.sowi.uni-mannheim.de/en/erdfelder/research/software/rrreg/

getPW

*Get Misclassification Matrices for RR Models*

Description

Given some randomization probabilities \( p \), each RR design corresponds to a misclassification matrix \( PW \). This square matrix has entries defined as: \( PW[i, j] = P(\text{respond } i | \text{ true state } j) \).

Usage

getPW(model, p, group = 1, par2 = NULL, Kukrep = 1)

Arguments

- model: one of the available models in the package RRreg;
- p: randomization probability;
- group: group index (1 or 2) for two-group designs such as "UQTunknown" or "SLD";
- par2: the second, estimated parameter in two-group designs (e.g., the unknown prevalence of the irrelevant question in "UQTunknown", the t-parameter for truth in the "SLD");
- Kukrep: number of replications in Kuk’s RR design (how many cards are drawn).
Details

The method is used internally for estimation. Moreover, the method might be useful to check the exact definition of the RR designs.

Note that for two-group designs, the matrix depends on a second parameter that is estimated from the data (e.g., the unknown prevalence of the unknown question in the unrelated question technique). Hence, the matrix itself is not constant, but an estimate of a random variable itself.

References


Examples

getPW(model = "Warner", p = 2/12)
getPW(model = "UQTknown", p = c(2/12, .3))
getPW(model = "UQTunknown", p = c(2/12, .10/12), group=2, par2=.4)

minarets

Minaret Data

Description

Data by Radukic and Musch (2014)

Usage

data(minarets)

Format

An object of class data.frame with 1621 rows and 7 columns.

Details

The following variables are included:

- `age` in years
- `leftRight` political left-right orientation on a scale from -5 to 5
- `rrt` response to RR question (SLD with randomization probabilities p=c(2/12,10/12))
- `condition` group membership in SLD (either randomization probability p[1] or p[2])
- `RRdesign` whether the respondent answered to the RR question (RRdesign=1) or to the direct question (RRdesign=-1)
- `leftRight.c` zero-centered political left-right orientation
- `age.c` zero-centered age
**plot.powerplot**  
*Plot power of multivariate RR methods*

**Description**
Plot estimated power from Monte Carlo simulation as a function of the sample size, separately for different effect sizes and multivariate RR methods.

**Usage**
```r
## S3 method for class 'powerplot'
plot(x, ...)
```

**Arguments**
- `x`: a *powerplot* object
- `...`: ignored

**plot.RRlog**  
*Plot Logistic RR Regression*

**Description**
Plot predicted logit values/probabilities of a randomized response logistic regression model.

**Usage**
```r
## S3 method for class 'RRlog'
plot(x, predictor = NULL,
     type = c("link", "response", "attribute"),
     center.preds = TRUE,
     plot.mean = TRUE,
     ci = 0.95,
     xlim = NULL,
     steps = 50,
     ...)
```
powerplot

Arguments

- **x**: a fitted RRlog object
- **predictor**: character name of a predictor of the model to be fitted
- **type**: "response" returns predicted probabilities for the (observable) RR responses, "link" returns predicted logit-values for the (latent) sensitive attribute, and "attribute" returns predicted probabilities of having the (latent) sensitive attribute.
- **center.preds**: whether to compute predictions by assuming that all other predictors are at their respective mean values (if FALSE: all other predictors are set to zero)
- **plot.mean**: whether to plot the mean of the predictor as a vertical line
- **ci**: level for confidence intervals. Use ci=0 to omit.
- **xlim**: if provided, these boundaries are used for the predictor on the x-axis
- **steps**: number of steps for plotting
- **...**: other arguments passed to the function plot (e.g., ylim=c(0,1)).

See Also

predict.RRlog

Examples

```r
# generate data
n <- 500
x <- data.frame(x1=rnorm(n))
pi.true <- 1/(1+exp(.3+1.5*x$x1))
true <- rbinom(n, 1, plogis(pi.true))
data <- RRgen(n, trueState=true, model="Warner", p=.1)
x$response <- data$response

# fit and plot RR logistic regression
mod <- RRlog(response ~ x1, data=x, model="Warner", p=.1)
plot(mod, "x1", ci=.95, type = "attribute", ylim = 0:1)
```

powerplot  

*Power plots for multivariate RR methods*

Description

Uses the function RRsimu to estimate the power of the multivariate RR methods (correlation RRcor, logistic regression RRlog, and/or linear regression RRlin).
Usage

powerplot(
  numRep,
  n = c(100, 500, 1000),
  pi,
  cor = c(0, 0.1, 0.3),
  b.log = NULL,
  model,
  p,
  method = c("RRcor", "RRlog", "RRlin"),
  complyRates = c(1, 1),
  sysBias = c(0, 0),
  groupRatio = 0.5,
  alpha = 0.05,
  nCPU = 1,
  show.messages = TRUE
)

Arguments

  numRep  number of bootstrap replications
  n       vector of samples sizes
  pi      true prevalence
  cor     vector of true correlations
  b.log   vector of true logistic regression coefficients
  model   randomized response model
  p       randomization probability
  method  multivariate RR method
  complyRates probability of compliance within carriers/noncarriers of sensitive attribute
  sysBias probability of responding 'yes' in case of noncompliance
  groupRatio ratio of subgroups in two-group RR designs
  alpha   type-I error used to estimate power
  nCPU    either the number of CPU cores or a cluster initialized via makeCluster.
  show.messages toggle printing of progress messages

Value

  a list of the class powerplot containing an array res with the power estimates and details of the simulation (e.g., model, p, pi, etc.)

See Also

  RRsimu for Monte-Carlo simulation / parametric bootstrap
Examples

# Not run
# pplot <- powerplot(100, n=c(150,250), cor=c(0,.3,.5),
# method="RRlog", pi=.6, model="Warner", p=.3)
# plot(pplot)

predict.RRlog  
Predict Individual Prevalences of the RR Attribute

Description

Predictions of the RR logistic regression model for the individual probabilities (or logits) of having the sensitive RR attribute, or of the probability of the RR responses.

Usage

## S3 method for class 'RRlog'
predict(
  object,
  newdata = NULL,
  type = c("link", "response", "attribute"),
  se.fit = FALSE,
  ci = 0.95,
  ...)

Arguments

object  
A fitted RRlog model

newdata  
An optional vector, matrix, or data.frame with values on the predictor variables. Note that for matrices, the order of predictors should match the order of predictors in the formula. Uses the fitted values of the model if omitted.

type  
"response" returns predicted probabilities for the (observable) RR responses, "link" returns predicted logit-values for the (latent) sensitive attribute, and "attribute" returns predicted probabilities of having the (latent) sensitive attribute.

se.fit  
Return standard errors for the predicted values in addition to confidence intervals. SEs on the logit scale are computed using the observed Fisher information from the fitted model. Standard errors for the probability scale are computed using the delta method.

ci  
Confidence level for confidence interval. If ci=FALSE, no confidence interval is returned. Confidence intervals on the probability scale (if type="response" or type="attribute") are computed based on the CI on the logit-scale using the inverse link function (hence, the CI will in general not be symmetric).

...  
ignored
RRcor

Value

either a vector of predicted values or a matrix with columns for the point estimates, confidence
interval, and standard errors (if se.fit=TRUE and ci=.95).

RRcor  Bivariate correlations including randomized response variables

Description

RRcor calculates bivariate Pearson correlations of variables measured with or without RR.

Usage

RRcor(
  x,
  y = NULL,
  models,
  p.list,
  group = NULL,
  bs.n = 0,
  bs.type = c("se.n", "se.p", "pval"),
  nCPU = 1
)

Arguments

x a numeric vector, matrix or data frame.
y NULL (default) or a vector, matrix or data frame with compatible dimensions to x.
models a vector defining which RR design is used for each variable. Must be in the same
order as variables appear in x and y (by columns). Available discrete models:
Warner, Kuk, FR, Mangat, UQTknown, UQTunknown, Crosswise, Triangular,
SLD and direct (i.e., no randomized response design). Available continuous
models: mix.norm, mix.exp.
p.list a list containing the randomization probabilities of the RR models defined in
models. Either, all direct-variables (i.e., no randomized response) in models
can be excluded in p.list; or, if specified, randomization probabilities p are
ignored for direct-variables. See RRruni for a detailed specification of p.
group a matrix defining the group membership of each participant (values 1 and 2)
for all multiple group models(SLD, UQTunknown). If only one of these models
is included in models, a vector can be used. For more than one model, each
column should contain one grouping variable
bs.n number of samples used to get bootstrapped standard errors
to get bootstrapped standard errors, use "se.p" for the parametric and/or "se.n" for the nonparametric bootstrap. Use "pval" to get p-values from the parametric bootstrap (assuming a true correlation of zero). Note that bs.n has to be larger than 0. The parametric bootstrap is based on the assumption, that the continuous variable is normally distributed within groups defined by the true state of the RR variable. For polytomous forced response (FR) designs, the RR variable is assumed to have equally spaced distances between categories (i.e., that it is interval scaled)

only relevant for the bootstrap: either the number of CPU cores or a cluster initialized via makeCluster.

Details

Correlations of RR variables are calculated by the method of Fox & Tracy (1984) by interpreting the variance induced by the RR procedure as uncorrelated measurement error. Since the error is independent, the correlation can be corrected to obtain an unbiased estimator.

Note that the continuous RR model mix.norm with the randomization parameter p=c(p.truth,mean,SD) assumes that participants respond either to the sensitive question with probability p.truth or otherwise to a known masking distribution with known mean and SD. The estimated correlation only depends on the mean and SD and does not require normality. However, the assumption of normality is used in the parametric bootstrap to obtain standard errors.

Value

RRcor returns a list with the following components:

r estimated correlation matrix
rSE.p, rSE.n standard errors from parametric/nonparametric bootstrap
prob two-sided p-values from parametric bootstrap
samples.p, samples.n sampled correlations from parametric/nonparametric bootstrap (for the standard errors)

References


See Also

data('RRreg') or http://www.dwheck.de/separate_content/RRregManual/index.html for a detailed description of the RR models and the appropriate definition of p

Examples

# generate first RR variable
n <- 1000
p1 <- c(.3,.7)
gData <- RRgen(n,pi=.3,model="Kuk",p1)
# generate second RR variable
p2 <- c(.8,.5)
t2 <- rbinom(n=n, size=1, prob=(gData$true+1)/2)
temp <- RRgen(model="UQTknown", p=p2, trueState=t2)
gData$UQTresp <- temp$response
gData$UQTtrue <- temp$true

# generate continuous covariate
gData$cov <- rnorm(n,0,4) + gData$UQTtrue + gData$true

# estimate correlations using directly measured / RR variables
cor(gData[,c("true","cov","UQTtrue")])
RRcor(x=gData[,c("response","cov","UQTresp")],
models=c("Kuk","d","UQTknown"),p.list= list(p1,p2) )

---

## RRgen

*Generate randomized response data*

### Description

The method `RRgen` generates data according to a specified RR model, e.g., "Warner". True states are either provided by a vector `trueState` or drawn randomly from a Bernoulli distribution. Useful for simulation and testing purposes, e.g., power analysis.

### Usage

```r
RRgen(
  n,
  pi.true,
  model,
  p,
  complyRates = c(1, 1),
  sysBias = c(0, 0),
  groupRatio = 0.5,
  Kukrep = 1,
  trueState = NULL
)
```

### Arguments

- `n`: sample size of generated data
- `pi.true`: true proportion in population (a vector for m-categorical "FR" or "custom" model)
- `p`: randomization probability (depending on model, see `RRruni` for details)
complyRates vector with two values giving the proportions of carriers and non-carriers who adhere to the instructions, respectively

sysBias probability of responding 'yes' (coded as 1) in case of non-compliance for carriers and non-carriers of the sensitive attribute, respectively. If sysBias=c(0,0), carriers and non-carriers systematically give the nonsensitive response 'no' (also known as self-protective(SP)-'no' responses). If sysBias=c(0,0.5), carriers always respond 'no' whereas non-carriers randomly select a response category. Note that sysBias = c(0.5, 0.5) might be the best choice for Kuk and Crosswise. For the m-categorical "FR" or "custom" model, sysBias can be given as a probability vector for categories 0 to (m-1).

groupRatio proportion of participants in group 1. Only required for two-group models, e.g., SLD and CDM

Kukrep Number of repetitions of Kuk's procedure (how often red and black cards are drawn)

trueState optional vector containing true states of participants (i.e., 1 for carriers and 0 for noncarriers of sensitive attribute; for FR: values from 0,1,...,M-1 (M = number of response categories) which will be randomized according to the defined procedure (if specified, n and pi.true are ignored)

Details

If trueState is specified, the randomized response procedure will be simulated for this vector, otherwise a random vector of length n with true proportion pi.true is drawn. Respondents answer biases can be simulated by adjusting the compliance rates: if complyRates is set to c(1,1), all respondents adhere to the randomization procedure. If one or both rates are smaller than 1, sysBias determines whether noncompliant respondents systematically choose the nonsensitive category or whether they answer randomly.

SLD - to generate data according to the stochastic lie detector with the proportion t of honest carriers, parameters are set to complyRates=c(t,1) and sysBias=c(0,0)

CDM - to generate data according to the cheating detection model with the proportion gamma of cheaters, parameters are set to complyRates=c(1-gamma,1-gamma) and sysBias=c(0,0)

Value
data.frame including the variables true and response (and for SLD and CDM a third variable group)

See Also

see vignette('RRreg') for a detailed description of the models and RRlog, RRlin and RRcor for the multivariate analysis of RR data

Examples

# Generate responses of 1000 people according to Warner's model,
# every participant complies to the RR procedure
genData <- RRgen(n=1000, pi.true=.3, model="Warner", p=.7)
colMeans(genData)
# use Kuk's model with two decks of cards,
# p gives the proportions of red cards for carriers/noncarriers
genData <- RRgen(n=1000, pi.true=.4, model="Kuk", p=c(.4,.7))
colMeans(genData)

# Stochastic Lie Detector (SLD):
# Only 80% of carriers answer according to the RR procedure
genData <- RRgen(n=1000, pi.true=.2, model="SLD", p=c(.2,.8),
    complyRates=c(.8,1),sysBias=c(0,0))
colMeans(genData)

## RRlin

### Linear randomized response regression

**Description**

Linear regression for a continuous criterion, using randomized-response (RR) variables as predictors.

**Usage**

RRlin(
    formula,
    data,
    models,
    p.list,
    group = NULL,
    Kukrep = 1,
    bs.n = 0,
    nCPU = 1,
    maxit = 1000,
    fit.n = 3,
    pibeta = 0.05
)

**Arguments**

- **formula**: a continuous criterion is predicted by one or more categorical RR variables defined by `models`. If the number of predictors exceeds the number defined by the vector `models`, the remaining predictors are treated as non-randomized variables (e.g., direct questions). Interactions including any of the RR variables cannot be included.

- **data**: an optional data frame, list or environment, containing the variables in the model.

of \( p \) list, where the entry \( p[i,j] \) defines the probability of responding \( i \) (i-th row) given a true state of \( j \) (j-th column)).

- **p.list**: list of randomization probabilities for RR models in the same order as specified in models. Note, that the randomization probabilities \( p \) must be provided in a list, e.g., `list(p=c(.2,.3))`. See `RRuni` for details.

- **group**: vector or matrix specifying group membership by the indices 1 and 2. Only for multigroup RR models, e.g., UQTunknown, CDM or SLD

- **Kukrep**: defines the number of repetitions in Kuk’s card playing method

- **bs.n**: Number of samples used for the non-parametric bootstrap

- **nCPU**: only relevant for the bootstrap: either the number of CPU cores or a cluster initialized via `makeCluster`.

- **maxit**: maximum number of iterations in optimization routine

- **fit.n**: number of fitting runs with random starting values

- **pibeta**: approximate ratio of probabilities \( p \) to regression weights \( \beta \) (to adjust scaling). Can be used for speeding-up and fine-tuning ML estimation (i.e., choosing a smaller value for larger \( \beta \) values).

### Value

Returns an object `RRlin` which can be analysed by the generic method `summary`.

### Author(s)

Daniel W. Heck

### References


### See Also

- vignette('RRreg') or [http://www.dwheck.de/separate_content/RRregManual/index.html](http://www.dwheck.de/separate_content/RRregManual/index.html) for a detailed description of the RR models and the appropriate definition of \( p \)

### Examples

```r
# generate two RR predictors
dat <- RRgen(n=500, pi=.4, model="Warner", p=.3)
dat2 <- RRgen(n=500, pi=c(.4,.6), model="FR", p=c(.1,.15))
dat$FR <- dat2$response
dat$trueFR <- dat2$true

# generate a third predictor and continuous dependent variables
dat$nonRR <- rnorm(500, 5, 1)
dat$depvar <- 2*dat$true - 3*dat2$true + .5*dat$nonRR + rnorm(500, 1, 7)
```
# use RRlin and compare to regression on non-RR variables
linreg <- RRlin(depvar~response+FR+nonRR, data=dat,
models=c("Warner","FR"),
  p.list=list(.3, c(.1,.15)), fit.n=1)
summary(linreg)
summary(lm(depvar~true +trueFR+nonRR, data=dat))

RRlog

Logistic randomized response regression

Description
A dichotomous variable, measured once or more per person by a randomized response method, serves as dependent variable using one or more continuous and/or categorical predictors.

Usage
RRlog(
  formula, data, model, p, group, n.response = 1, LR.test = TRUE, fit.n = 3, EM.max = 1000, optim.max = 500, ...
)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>formula</td>
<td>specifying the regression model, see formula</td>
</tr>
<tr>
<td>data</td>
<td>data.frame, in which variables can be found (optional)</td>
</tr>
<tr>
<td>p</td>
<td>randomization probability/probabilities (depending on model, see RRuni for details)</td>
</tr>
<tr>
<td>group</td>
<td>vector specifying group membership. Can be omitted for single-group RR designs (e.g., Warner). For two-group RR designs (e.g., CDM or SLD), use 1 and 2 to indicate the group membership, matching the respective randomization probabilities p[1],p[2]. If an RR design and a direct question (DQ) were both used in the study, the group indices are set to 0 (DQ) and 1 (RR; 1 or 2 for two-group designs).</td>
</tr>
</tbody>
</table>

RRlog
RR designs). This can be used to test, whether the RR design leads to a different prevalence estimate by including a dummy variable for the question format (RR vs. DQ) as predictor. If the corresponding regression coefficient is significant, the prevalence estimates differ between RR and DQ. Similarly, interaction hypotheses can be tested (e.g., the correlation between a sensitive attribute and a predictor is only found using the RR but not the DQ design). Hypotheses like this can be tested by including the interaction of the DQ-RR-dummy variable and the predictor in formula (e.g., RR ~ dummy*predictor).

- **n.response**: number of responses per participant, e.g., if a participant responds to 5 RR questions with the same randomization probability \( p \) (either a single number if all participants give the same number of responses or a vector)
- **LR.test**: test regression coefficients by a likelihood ratio test, i.e., fitting the model repeatedly while excluding one parameter at a time (each nested model is fitted only once, which can result in local maxima). The likelihood-ratio test statistic \( G^2(\text{df}=1) \) is reported in the table of coefficients as deltaG2.
- **fit.n**: Number of fitting replications using random starting values to avoid local maxima
- **EM.max**: maximum number of iterations of the EM algorithm. If EM.max=0, the EM algorithm is skipped.
- **optim.max**: Maximum number of iterations within each run of optim

**Details**

The logistic regression model is fitted first by an EM algorithm, in which the dependend RR variable is treated as a misclassified binary variable (Magder & Hughes, 1997). The results are used as starting values for a Newton-Raphson based optimization by optim.

**Value**

Returns an object RRlog which can be analysed by the generic method summary. In the table of coefficients, the column Wald refers to the Chi^2 test statistic which is computed as Chi^2 = z^2 = Estimate^2/StdErr^2. If LR.test = TRUE, the test statistic deltaG2 is the likelihood-ratio-test statistic, which is computed by fitting a nested logistic model without the corresponding predictor.

**Author(s)**

Daniel W. Heck

**References**


**See Also**

vignette('RRreg') or http://www.dwheck.de/separate_content/RRregManual/index.html for a detailed description of the RR models and the appropriate definition of \( p \)
Examples

```r
# generate data set without biases
dat <- RRgen(1000, pi=.3, "Warner", p=.9)
dat$covariate <- rnorm(1000)
dat$covariate[dat$true==1] <- rnorm(sum(dat$true==1), .4, 1)
# analyse
ana <- RRlog(response~covariate, dat, "Warner", p=.9, fit.n = 1)
summary(ana)
# check with true, latent states:
glm(true~covariate, dat, family=binomial(link="logit"))
```

---

**Description**

Uses the package *lme4* to fit a generalized linear mixed model (GLMM) with an adjusted link function.

**Usage**

```r
RRmixed(formula, data, model, p, const = 1e-04, adjust_control = FALSE, ...)
```

**Arguments**

- `formula`: two-sided formula including random and fixed effects (see below or `glmer` for details)
- `data`: an optional data frame with variables named in formula
- `model`: type of RR design. Only 1-group RR designs are supported at the moment (i.e., "Warner", "FR", "UQTknown", "Crosswise", "Triangular", "Kuk", "Mangat", "custom"). See `RRuni` or vignette(`RRreg`) for details.
- `p`: randomization probability
- `const`: the RR link function is not defined for small and/or large probabilities (the boundaries depend on model and p). To increase robustness of the estimation, these probabilities smaller and larger than the respective boundaries (plus/minus a constant defined via `const`) are smoothed by separate logit-link functions.
- `adjust_control`: whether to adjust the control arguments for `glmer`, which might help in case of convergence issues for some models. `lmerControl` settings are changed to `nAGQ=0initStep = FALSE` and optimizer = "bobyqa".
- `...`: further arguments passed to `glmer`
Details

Some examples for formula:

- random intercept: \( \text{response} \sim \text{covariate} + (1 \mid \text{group}) \)
- random slope: \( \text{response} \sim \text{covariate} + (0 + \text{covariate} \mid \text{group}) \)
- both random slope and intercept: \( \text{response} \sim \text{covariate} + (\text{covariate} \mid \text{group}) \)
- level-2 predictor (must have constant values within groups!): \( \text{response} \sim \text{lev2} + (1 \mid \text{group}) \)

Note that parameter estimation will be unstable and might fail if the observed responses are not in line with the model. For instance, a Forced-Response model (model=“FR”) with \( p = c(0,1/4) \) requires that expected probabilities for responses are in the interval \([.25,1.00]\). If the observed proportion of responses is very low («.25), intercepts will be estimated to be very small («0) and/or parameter estimation might fail. See \texttt{glmer} for setting better starting values and \texttt{lmerControl} for further options to increase stability.

Value

an object of class \texttt{glmerMod}

References


Examples

# generate data with a level-1 predictor
set.seed(1234)
d <- data.frame(group=factor(rep(LETTERS[1:20],each=50)),
cov=rnorm(20*50))

# generate dependent data based on logistic model (random intercept):
d$true <- simulate(~ cov + (1|group), newdata=d,
family=binomial(link="logit"),
newparams=list(beta=c("(Intercept)"=-.5, cov=1),
theta=c("group.(Intercept)"=1)))[[1]]

# scramble responses using RR:
model <- "FR"
p <- c(true0=.1, true1=.2)
d$resp <- RRgen(model="FR", p=p, trueState=d$true)$response

# fit model:
mod <- RRmixed(resp ~ cov + (1|group), data=d, model="FR", p=p)
summary(mod)
Monte Carlo simulation for one or two RR variables

Description

Simulate and analyse bivariate data including either one RR variable (either correlation, logistic, or linear regression model) or two RR variables (only correlations). Useful for power analysis, parametric bootstraps or for testing the effects of noncompliance on the stability of estimates.

Usage

RRsimu(
  numRep,
  n,
  pi,
  model,
  p,
  cor = 0,
  b.log = 0,
  complyRates = c(1, 1),
  sysBias = c(0, 0),
  method = c("RRuni", "RRcor", "RRlog", "RRlin"),
  alpha = 0.05,
  groupRatio = 0.5,
  MLeSt = FALSE,
  getPower = TRUE,
  nCPU = 1
)

Arguments

numRep  number of replications
n       sample size
pi      true proportion of carriers of sensitive attribute (for 2 RR variables: vector)
model   either one or two RR model (as vector), see RRuni
p       randomization probability (for 2 RR variables: a list). See RRuni for details.
cor     true Pearson-correlation used for data generation (for RRcor). Can also be used to generate data with two dichotomous RR variables.
b.log   true regression coefficient in logistic regression (for RRlog)
complyRates vector with two values giving the proportions of participants who adhere to the instructions in the subset with or without the sensitive attribute, respectively (for 2 RR variables: a list)
sysBias probability of responding 'yes' (coded as 1 in the RR variable) in case of noncompliance for carriers and non-carriers, respectively. See RRgen
RRsimu

method vector specifying which RR methods to be used in each replication. For a single RR variable, the methods RRuni, RRcor, RRlog, and RRlin are available. For 2 RR variables, only RRcor is available.

alpha significance threshold for testing the logistic regression parameter beta

groupRatio proportion of participants in group 1. Only for two-group models (e.g., "SLD") (for 2 RR variables: vector)

MLest concerns RRuni: whether to use optim to get ML instead of moment estimates (only relevant if pi is outside of [0,1])

getPower whether to compute power for method="RRcor" (performs an additional bootstrap assuming independence)

nCPU either the number of CPU cores or a cluster initialized via makeCluster.

Details

For a single RR variable:

The parameter b_log is the slope-coefficient for the true, latent values in a logistic regression model that is used for data generation.

The argument cor is used for data generation for linear models. The directly measured covariate is sampled from a normal distribution with shifted means, depending on the true state on the sensitive attribute (i.e., the true, underlying values on the RR variable). For dichotomous RR variables, this corresponds to the assumption of an ordinary t-test, where the dependent variable is normally distributed within groups with equal variance. The difference in means is chosen in a way, to obtain the point-biserial correlation defined by cor.

For two RR variables:

cor has to be used. In case of two dichotomous RR variables, the true group membership of individuals is sampled from a 2x2 cross table. Within this table, probabilities are chosen in a way, to obtain the point-tetrachoric correlation defined by cor

Note, that for the FR model with multiple response categories (e.g., from 0 to 4), the specified cor is not the exact target of the sampling procedure. It assumes a normal distribution for each true state, with constant differences between the groups (i.e., it assumes an interval scaled variable).

Value

A list containing

parEsts matrix containing the estimated parameters

results matrix with mean parameters, standard errors, and number of samples to which the respective method could not be fitted

power vector with the estimated power of the selected randomized response procedures

Examples

# Not run: Simulate data according to the Warner model
# mcsim <- RRsimu(numRep=100, n=300, pi=.4,
#                  model="Warner", p=.2, cor=.3)
# print(mcsim)
**RRuni**  
*Univariate analysis of randomized response data*

**Description**

Analyse a data vector `response` with a specified RR model (e.g., Warner) with known randomization probability `p`

**Usage**

```r
RRuni(response, data, model, p, group = NULL, MLest = TRUE, Kukrep = 1)
```

**Arguments**

- `response` either vector of responses containing 0='no' and 1='yes' or name of response variable in `data`. For the Forced Response (FR) model, response values are integers from 0 to (m-1), where 'm' is the number of response categories. In Kuk's card playing method (Kuk), the observed response variable gives the number of red cards.
- `data` optional `data.frame` containing the response variable
- `p` randomization probability (see details or vignette("RRreg"))
- `group` a group vector of the same length as `response` containing values 1 or 2, only required for two-group models, which specify different randomization probabilities for two groups, e.g., CDM or SLD. If a `data.frame` `data` is provided, the variable group is searched within it.
- `MLest` whether to use `optim` to get ML instead of moment estimates (only relevant if `pi` is outside of [0,1])
- `Kukrep` number of repetitions of Kuk's card-drawing method

**Details**

Each RR design model differs in the definition of the randomization probability `p`, which is defined as a single probability for

- "Warner": Probability to get sensitive Question
- "Mangat": Prob. for noncarriers to respond truthfully (i.e., with No=0)
- "Crosswise": Probability to respond 'yes' to irrelevant second question (coding of responses: 1=['no-no' or 'yes-yes']; 0=['yes-no' or 'no-yes'])
- "Triangular": Probability to respond 'yes' to irrelevant second question (coding of responses: 0='no' to both questions (='circle'); 1='yes' to at least one question ('triangle'))
and as a two-valued vector of probabilities for

- "Kuk": Probability of red cards in first and second set, respectively (red=1, black=0);
- Unrelated Question ("UQTknown"): Prob. to respond to sensitive question and known prevalence of 'yes' responses to unrelated question
- Unrelated Question ("UQTunknown"): Prob. to respond to sensitive question in group 1 and 2, respectively
- Cheating Detection ("CDM"): Prob. to be prompted to say yes in group 1 and 2, respectively
- Symmetric CDM ("CDMsym"): 4-valued vector: Prob. to be prompted to say 'yes'/no' in group 1 and 'yes'/no' in group 2
- Stochastic Lie Detector ("SLD"): Prob. for noncarriers to reply with 0='no' in group 1 and 2, respectively
- Forced Response model ("FR"): m-valued vector (m=number of response categories) with the probabilities of being prompted to select response categories 0,1,...,m-1, respectively (requires \( \sum(p)<1 \))
- RR as misclassification ("custom"): a quadratic misclassification matrix is specified, where the entry \( p[i,j] \) defines the probability of responding i (i-th row) given a true state of j (j-th column) (see getPW)

For the continuous RR models:

- "mix.norm": 3-valued vector - Prob. to respond to sensitive question and mean and SD of the masking normal distribution of the unrelated question
- "mix.exp": 2-valued vector - Prob. to respond to sensitive question and mean of the masking exponential distribution of the unrelated question
- "mix.unknown": 2-valued vector - Prob. of responding to sensitive question in group 1 and 2, respectively

Value

an RRuni object, can be analyzed by using summary

See Also

text(quote(Var.RRreg)) or [http://www.dwheck.de/separate_content/RRregManual/index.html](http://www.dwheck.de/separate_content/RRregManual/index.html) for a detailed description of the RR models and the appropriate definition of p

Examples

# Generate responses of 1000 people according to Warner's model
# with an underlying true proportion of .3
genData <- RRgen(n=1000, pi=.3, model="Warner", p=.7)
# Analyse univariate data to estimate 'pi'
analyse <- RRuni(response=genData$response, model="Warner", p=.7)
summary(analyse)

# Generate data in line with the Stochastic Lie Detector
# assuming that 90% of the respondents answer truthfully
genData2 <- RRgen(n=1000, pi=.3, model="SLD", p=c(.2,.8), complyRates=c(.8,1), groupRatio=0.4)
analyse2 <- RRuni(response=genData2$response, model="SLD", p=c(.2,.8), group=genData2$group)
summary(analyse2)
Index

∗Topic datasets
  minarets, 4
∗Topic package
  RRreg-package, 2

formula, 15
getPW, 3, 22
glmer, 17, 18
list, 14
lme4, 17
lmerControl, 17, 18
makeCluster, 7, 10, 14, 20
minarets, 4
optim, 16
plot, 6
plot.powerplot, 5
plot.RRlog, 5
powerplot, 5, 6
predict.RRlog, 6, 8

RRcor, 2, 6, 9, 12, 19, 20
RRgen, 2, 11, 19
RRlin, 2, 6, 12, 13, 20
RRlog, 2, 6, 8, 12, 15, 19, 20
RRmixed, 17
RRreg(RRreg-package), 2
RRreg-package, 2
RRsimu, 2, 6, 7, 19
RRuni, 2, 9, 11, 14, 15, 17, 19, 20, 21

summary, 14, 16, 22